www.icgst.com

GA-based Parameter Optimization for Word Segmentation

Ammar Mohammed
Department of Computer Science, ISSR, Cairo University
Department of Computer Science, Arab East Colleges, Riyadh, KSA
ammar@cu.edu.eg,
Mohammed Karam, Hesham Hefny
Department of Computer Science, ISSR,Cairo University
Mohamed.Karam.Ali@pg.cu.edu.eg, hehefny@ieee.org ,

Abstract

word segmentation is the process of finding the best likely sequence of words from a sequence of concatenated characters without spaces. Several researches proposed solutions to word segmentation using heuristic methods. The main task of the last methods is to hopefully find the best segmentation without searching the entire state spaces. This paper proposes a new approach for word segmentation based on parameters optimization by means of Genetic Algorithm. The approach is tested on English language using two different language models taking into consideration several test sets. To show that the presented approach is domain language independent, the approach is experimented furthermore on the Arabic language. The experiments show that segmentation using parameters optimization gives better results.

Keywords: *NLP, Word segmentation, GA*

Nomenclature

NLP	Natural Language processing
GA	Genetic Algorithms
BNC	British National Corpus
N	Number of words per candidate solution
T	Trade-off parameter

1 Introduction

Word segmentation is the process of determining the spaces positions for a sequence of words without spaces, or it may also mean determining the morphemes of a word like the segmentation of the word "unreachable" to morphemes "un,reach,able". A human can segment words easily because of accumulative knowledge. However, this process is complicated for the machine because it does not have enough knowledge to deal with the ambiguity of the human languages.

The word segmentation methods depend on a raw list of words or on a language model. Those methods that depend on a raw list of words are called dictionary-based methods [6, 35, 26]. They use a heuristic functions that search locally for the words. Because of containing each local candidate solution, those dictionary-based methods usually have a low accuracy and low complexity. Generally, dictionary-based methods would be inefficient to segment most of the languages [17]. On the other hand, the methods that use language models are called statistical-based methods [33, 36]. The later methods are more accurate but highly complex. Norvig in [25] tried to solve the complexity problem by using a local search function that excludes all words with a length bigger than twenty characters. It is worth noting that the accuracy of those statistical methods depends on the size of the corpus that constructs the language model.

Hybrid methods can be constructed by integrating dictionary-based and statistical-based methods to enhance the complexity and accuracy. Our previous work in [23], proposed a hybrid approach for word segmentation that uses a local search technique for word segmentation. In this approach, the segmentation is built through an iterative process that separates the first word of the local candidate of N words with the best score, where N is fixed number estimated heuristically. We also, used a fitness function that depends on both the probability and the length of the words to increase the accuracy.

Although hybrid approaches [17, 38, 16, 29] tried to enhance the performance of the segmentation in terms of accuracy and the complexity, there are still, however, two main problems facing the word segmentation process. The first problem is the task of determining the local search space size. The previous task has impact on the accuracy and complexity. On the other hand, the second problem concerns with the

ambiguity problem found due to the possible unfair probability distribution in the language model when using a relatively small corpora. In this case the pure statistical methods will not be efficient to handle such situation. Teahan in [37] and Hockenmaier in [14] indicated that the corpus size greatly affects the performance of the segmentation system.

Introducing a new approach for word segmentation to handle the previous two problems is highly desirable. Particularly, estimating the parameters that affect the two problems will influence the accuracy of the segmentation process. The first parameter is choosing the length N of the local candidate segmentation. The second parameter is a trade-off weight between the length and the probability of the word. The latter parameter can help in solving the unfair probability distribution problem. To estimate those parameters we use a genetic-based approach for parameter optimization.

To this end, The main contribution of this paper is to propose a new approach for word segmentation containing a preprocessing step using the genetic algorithm to optimize segmentation parameters. The paper is doing so by extending and enhancing our previous work, found in [23], by estimating the previously mentioned two parameters. Additionally, the approach is tested on several experiments and the results are compared before and after parameter to measure the performance. The experiments run on both Google n-gram and BNC language models. Several data sets are used in the experiments. Furthermore, the proposed approach is tested on English and Arabic languages to show that our approach is language independent. Also, the experimental results of relatively small size of the Arabic data set show that the ability of the proposed approach to deal with the small-size corpora.

The reset of this paper is organized as the following; section 2 discusses some basic terminologies and background. Section 3 discusses the related work. Section 4 presents the proposed algorithm and the optimization method. Section 5 experiments the proposed approach on English and Arabic languages. Finally, Section 6 concludes the paper.

2 Background

This section introduces some terminologies on the statistical-based word segmentation methods [25] as well as performance measurements [17, 23]. Additionally, the section gives background on the genetic algorithms [12].

2.1 Word Segmentation

A computerized word segmentation can be used in different domains of applications. For example, word segmentation is an essential process in optical character recognition [5], where any incorrect segmentation of the scanned documents leads to errors in the information retrieval of the document. In NLP tasks, in particular speech recognition [30], word segmentation can also be used to identify the pauses in the speech [3]. Moreover, word segmentation can be used to identify words in those languages that are written without spaces (e.g., Japanese, Chinese, and Thai). In the previous types of languages, the words are not delimited by whitespace but, they must be inferred using the main characters sequence[10]. Word segmentation additionally can be used in databases schema matching to solve the semantic heterogeneity [21].

Broadly speaking, one of the most important aspects of the word segmentation system is the data captured from the human in the form of training data. Since the training data is huge, it can not be used directly in the process of the segmentation, but firstly it should be preprocessed and summarized [25]. The summary contains the words and their frequencies which give us the so-called *unigram*. Additionally, the summary might contain each two consecutive words and their frequencies to form the so-called *bigram*.This process can be generalized iteratively to form the so-called *n-gram*.

After extracting the *unigram*, the entries can be converted to a language model where the frequencies are replaced by the probabilities of the word. The probability P of a word w in the corpus is given as $P(w) = \frac{count\ (w)}{count(words)}$, where $count(words)$ is the count of all words in the corpus. Generally, in the *n-gram*, the frequency is replaced by the conditional probability $P(W_n|W_{n-1})$ of w_n given a previous word w_{n-1}, where $P(W_n|W_{n-1}) = \frac{count\ (W_{n-1}W_n)}{count\ (W_{n-1})}$. The probability of a sequence of n words $w_1, w_2, ..., w_n$ is given as $p(w_1w_2...w_n) = \prod_{i=1}^{n} w_i$. The *n-gram*,language model is the model in which the probability of a word only depends on the previous n words. For any *unseen word* w that is not contained in the language model, its probability is $P(w) = \frac{1}{count(Words)\ *\ (10^n)}$ [25]. Sometimes a zero frequency of words occurs in the *n-gram*, model, to cope with this problem, back-off process is used. In the later process, if there is no $n-gram$ for certain word sequence, it looks for a $(n-1)-gram$ instead.

Generally the performance of a word segmentation system is usually measured in terms of *precision (p)* and *recall (r)* and *F-measure* (f) [17, 28]. Informally, the term *precision (p)* indicates the percentage of the correctly segmented words to the output words of the system. The term *recall (r)* represents the percentage of the correctly segmented words to the number of words in the original text. Whereas the term *F-measure* is the harmonic mean of *precision (p)* and *recall (r)*. To express the previous measurements in other words, let N_1 expresses the true number of word boundaries in some original text, N_2


expresses the number of spaces between words of the output of a system, and N_3 expresses the number of correctly segmented output words of the system. The precision $p = \frac{N_3}{N_2}$, the recall $r = \frac{N_3}{N_1}$, and F-measure: $f = \frac{2.precision.recall}{precision+recall}$.

2.2 Genetic Algorithms

The term optimization is the process of making something (as a design, system, or decision) fully perfect, functional, or effective as possible. Usually, the optimization process includes an objective function with related parameters. There are several optimization methods existing. Genetic Algorithm (GA) is considered one of those successful methods to handle optimization problems [34, 22]. GA has higher ability to discover global optima and to solve multi-objective optimization problems. Additionally GA has the ability to deal with noisy functions well and to handle large and complex search spaces easily. Broadly speaking, GA was developed for the first time in [15]. GA are used for optimization in different fields. In [32] GA is used for the optimization of fingerprint recognition. In [39] GA optimizes the parameters of SVR system to forecast the volume of sales. [13] used the GA for Least-cost Design of water distribution Networks.

The idea of GA is inspired by the evolution theory of the species. In this theory, weak species are subjected to extinction by natural selection. On the other hand, strong species have a highly opportunity to move their genes to future generations by means of reproduction. Species holding the fitting combination of genes become dominant in the population in the long run. Also, during the evolution process, it might happen changes in genes. If these changes provide additional advantages for Species to survive, new species evolve from the old ones.

In GA terminology, a solution vector $x = [x_1, x_2, ..., x_N]$ where $x \in \mathcal{X}^N$ of N features is called a chromosome or an individual. Chromosomes consist of N units called genes. Where Each genome governs one or more features of the chromosome. Typically, a chromosome represents a single solution x in the solution space. Each genome represents an encoded value which is used in the evaluation process [4]. If the solution values in encoded form in the genome, then they are called genotype otherwise, they are called Phenotype [34]. Once all chromosomes of the current generation are evaluated, the selection process takes place. In the later process, the best-fitted chromosomes have a better chance to be selected to be parents to crossover and mutate [34]. The crossover represents the process of generating a new chromosome that has some characteristics of both parents, while mutation means changing in one or more of the features of the chromosome.

Generally, GA is equipped with parameters. These parameters can be classified into two types [2]. The first type is structural parameters that affect the structure of GA applications. Examples for this type of parameters include the stopping criterion, encoding schemes and the selection method. The second type, on the other hand, is the numerical parameters. This type includes the number of generations, the number of chromosomes per generation, mutation probability and crossover probability. There are existing two main approaches for parameter selection. The first approach is the empirical method which depends on sensitivity analysis. The second approach is the adaptive method which uses an initial parameter setting that is optimized while the algorithm is executed[27].

3 Related Work

This section discusses several related work of word segmentation. It begins first by discussing the traditional approaches for word segmentation followed by those related work that use genetic algorithms.

3.1 Traditional Approaches

Word segmentation methods can be classified into three categories;namely dictionary-based, statistical-based and hybrid-based methods. The category of dictionary-based methods generally depend on the so-called matching algorithm and list of words. Examples of those methods belonging to this category are Maximum Matching (MM), greedy matching and reverse MM [6]. The second category of word segmentation is the statistical-based methods. The latter methods depend on word statistics. The statistics of words are computed using the frequency of the words in some corpus. The work in [25], for example, solved the global search problem by using a local search algorithm. The previous work proposed a method that relies on the independence assumption and word maximum length. The work in [31] is another example for the statistical-based methods that depends on the frequency of syllabus instead of depending on words frequencies. Finally, the last category of word segmentation methods is hybrid approach methods. In these methods, both dictionary and statistical-based methods are combined in the same framework [23, 20, 40, 29, 17]. Our Previous work in [23] is considered an example of the word segmentation belonging to this category, in which we used a matching algorithm that matches three consecutive words. Additionally, we used a score function that depends on both probability and length of the three consecutive words.

In contrast to the work of this paper, the work of the previous three categories do not take into considerations optimization of the segmentation parameters.


3.2 Genetic Approaches

Introducing GA to word segmentation is not new. The work of [24] used GA to find an approximate global optimal segmentation from a set of possible segmentation for Vietnamese text. The experiments in this work showed that GA can give a considerable accuracy with less complexity. In contrast to the work of this paper, this method, however, is limited to Vietnamese texts, as it depends on the property of multi-syllable words of Vietnamese text.

Several work used GA to morphemes segmentation. For example the work in [18] used GA in prefix-suffix word segmentation to segment words into morphemes. GA have been used to extract word segmentation rules from a list of words. It is noteworthy that, the performance of word segmentation process depends on the quality of this rules list. Another work for morphemes segmentation using GA is the work presented in [11].This work provided an unsupervised technique for morphemes segmentation in Spanish as an inflective language. Unlike the previous work, they did not use GA to extract rules, instead, they used it to find the morphemes segmentation with the highest fitness. Also, this work is limited to Spanish language. Another work for morphemes segmentation using GA is in [8]. This work used GA for morphemes segmentation with general purpose and a specifically designed evaluation function. Particularly, the work proposed a suffix-based evaluation function depending on a list of correctly segmented words.

Differently to the work of this paper, the previous GA works are limited only to morphemes segmentation and they can not be applied in segmentation of a sequence of characters without spaces into meaningful words. In addition, these works are domain language dependent.

4 Proposed Work

This section proposes a method that enhances our previous work [23]. The enhancement replaces the manually tuned parameters with GA optimized parameters. As it was mentioned previously, there are two main parameters that should be optimized before the algorithm starts. The first one is the number of words per candidate solution. We will refer to this parameter as N. While the second parameter is the trade-off weight between the effect of the length and probability on the fitness function. We will refer to this parameter as T. Figure 1 shows the work-flow of the proposed method. Firstly, the GA uses the dataset to optimize the parameters. Once the optimized parameters values are calculated, they are fed to the word segmentation algorithm.

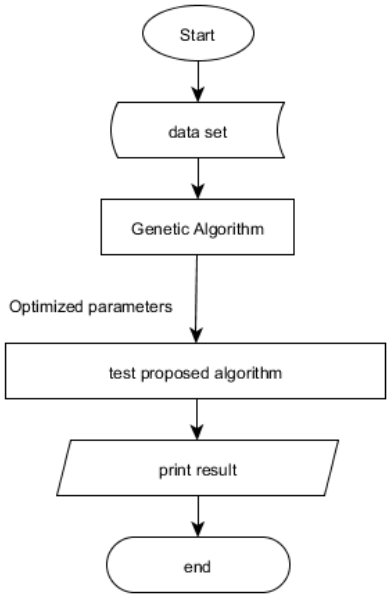


Figure 1: simple System architecture For the Parameter Optimization process Using Genetic Algorithms

4.1 The Word Segmentation Algorithm

Because of the importance of word probability and word length as they are used in both statistical-based and dictionary-based approaches, the majority of segmentation methods use the probability and word length as the fitness function. Figure 2 describes the proposed segmentation algorithm. The main functions that construct the segmentation algorithm are *segment(text)*, *firstOfBestCandidateSolution(inputString)*, *Match (inputstring)*, *candidateSolution(inputString)*, and *Pw(PreviousWord,word)*. The role of the function *segment(desegmented)* is to determine the boundaries of words from the input sequence of characters without spaces. It repeats the lines 3, line 4 and line 5. In line 3, it tries to find the best local candidate solution and separates its first word. Line 4 tries to add this word to the list of segmented words. Line 5 removes the letters of the first word from the unsegmented input. By repeating these three steps, the input string will be converted into a list of segmented words. The next few paragraphs illustrate these steps.

The function *firstOfBestCandidateSolutions(inputString)* returns the first word of the candidate solution which achieves the highest fitness. a candidate solution is N consecutive words that can be matched with the beginning of the input string. It is assumed that, the number of words per candidate solution was set before the algorithm starts. As it was said previously, GA is used to optimize this parameter.

The purpose of the function *CandidateSolution* is


to find N of consecutive words that can matches the beginning of the input string with their fitness. To match a single word, the function *Match (inputstring)* is used. This function returns a single word from the dictionary that must be matched with the beginning of the *inputtring*. Once a candidate solution is found, the fitness of this candidate solution is calculated, and a pair of *ListOfWords,Score* is returned. This Score is calculated based on probability and length of the word as depicted in 1. The last function is *Pw(PreviousWord,word)* is used to find the probability of a word.

$$Score = \frac{A}{B} \tag{1}$$

where,
$A = log(p(word_1| < S >)) + \sum_{i=1}^{N} log(p(word_{i+1}|word_i)$
and $B = \sum_{i=1}^{N} length(word_i))^T$

It should be mentioned that the proposed algorithm will not treat the problem of unseen words. This means that the unseen words will be wrongly segmented to seen words. From our perspective, the statistical methods suffer from the same problem. In most of the cases, these methods will segment the unseen word to seen words. For unseen characters or symbols, we will segment this character or symbol as a word.

4.2 Parameter Optimization

This section explains how the GA is used to optimize the two main parameters N and T of the segmentation algorithm. The optimization starts by splitting data set into three sets. The first set is the training set which is summarized into the language model. The second set is the development set which is used to compute the parameter values. The third set is the test set which is used to test the accuracy of the segmentation algorithm after parameter optimization.

Once the first step extracted the language model for the training data sets, the GA starts the process of optimization. An initial generation of chromosomes is created with random values. After that, the breeding process starts. The later process consists of the three steps namely selection, reproduction and replacement [34]. The selection step starts with the evaluation of the chromosomes of the entire generation. To evaluate the fitness of a single chromosome, the genotypes values of a genome will be converted into phenotypes which is used later as parameters for the segmentation algorithm. Then the F-measure is used to evaluate the fitness of each chromosome. Then, the random selection with a probability depending on the fitness takes place. In the reproduction step, which involves crossover and mutation, the creation of new chromosomes having some characteristic of their parents is produced. The final step of breeding is the replacement step in which the old chromosomes are killed and are replaced by the new generation of chromosomes. In this work, the uniform crossover is applied, and the mutation is represented by adding or subtracting random values. The process of breeding is repeated until all generations are processed.

At the end of GA process, the optimized parameters values are computed and then are given to the segmentation algorithm to test the performance of the algorithm on a given data set. Figure 3 illustrates the previous steps.

5 Experiments

This section examines the performance of the proposed approach by doing several the experimental using different language models and different test sets In both English and Arabic language. For the English language, we used the BNC[1] and Google N-gram[2] language models and set of corpora that included Brown, Inaugural, ABC, Shakespear and Gutneburg as test sets [3]. For the Arabic Language, we used Al-Watan dataset [1].

This section classifies the experiments into three categories based on the used language model. In the first category, the experiments run on google N-gram language model. In the second category, the experiments are performed in BNC language model. In the last category, the experiments run on Al-watan Language model. Each category starts with the GA parameter values then shows the performance results. The performance is compared to the results presented in [23]. In these experiments, the empirical approach is used to determine the parameters for the genetic algorithm[9].

5.1 Google N-gram Language Model Results

This section discusses the experiments of the proposed approach using Google N-gram language model. The genetic algorithm was experimented several times with different GA parameters settings with a development set that contains $200,000$ word. The settings that achieved the highest f-measure was selected. These settings were as the following; number of chromosomes per generation: 100, number of generations: 100, probability of crossover: 0.7, probability of mutation: 0.3 and selection strategy: Roulette Wheel with Elitism. The best results obtained from the genetic algorithm was 97.3% F-Measure when $N = 4$ and $T = 1.3$. Later, we used the optimized values of N and T to test the algorithm on the test sets. Table 1 shows the results of the pre without optimization, while table2 shows the results after performing the optimization.

[1]available at http://www.natcorp.ox.ac.uk

[2]available at http://norvig.com/ngrams/

[3]$Availableat : http : //www.nltk.org/nltk_data/$

```
1. FUNCTION Segment (desegmented)
2. WHILE(desegmented is not empty)
3.     currentWord = firstOfbestCandidateSolution(inputString)
4.     add currentWord to the list of segmented words
5.     Remove letters of currentWord from the beginning of inputString
6. ENDWHILE
7. Return wordlist
8. ENDFUNCTION

9. FUNCTION FirstOfBestCandidateSolution(inputString)
10. Find all Possible Candidate solutions that can match With the InputString
11. Result ⟵ first word from Candidate Solution that has the Maximum Score
12. Return Result
13. ENDFUNCTION

14. FUNCTION CandidateSolution(inputString)
15. SequenceOfChars = inputString
16. K ← 1
17. WHILE k <= N
18.     W_K = Match(SequenceOfChars)
19.     Remove letters of W_K from the beginning of SequenceOfChars
20.     add W_K to the ListOfCandidateWords
21. k   ← k + 1
22. ENDWHILE
23. calculate the score as equation. 1
24. Result ⟵ (ListOfCandidateWords,Score)
25. Return Result
26. ENDFUNCTION

27. FUNCTION match(inputString)
28. IF (Length(inputString) =0)
29.     Return < emptyString >
30. ENDIF
31. Result ⟵ a word that match the beining of inputstring
32. Return result
33. ENDFUNCTION

34. FUNCTION Pw(previousWord,word)
35. IF (word=< emptyString >)
36.     Return 1
37. ENDIF
38. Return the probability of word according to back-off concept
39. ENDFUNCTION
```

Figure 2: Proposed Method

By comparing the results before and after optimization we noticed that the F-measure is enhanced in all corpus.

5.2 BNC Language Model Results

This section discusses the experiments of the proposed method using BNC language model. The genetic algorithm was experimented several times with different GA parameters settings with a development set that contains 200,000 word. The settings that achieved the highest f-measure was selected. These settings

Table 1: The results of the algorithm without optimization from work [23]

	Recall	Precision	F-measure	Size of test set
Brown	96.54%	94.63%	95.57 %	1003881
Inaugural	98.71%	98.76%	98.74 %	130271
ABC	97.31%	96.66%	96.98%	630004
Shakespear	93.97%	92.43%	93.2%	184194
Gutenburg	96.91%	96.53%	96.72%	1942398

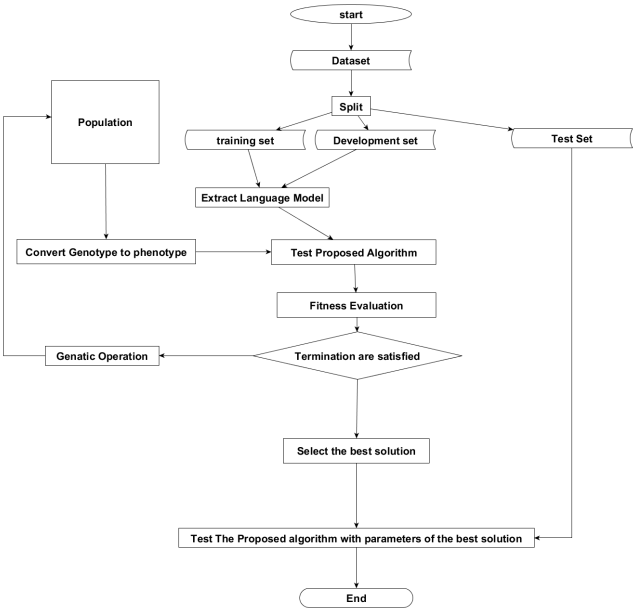


Figure 3: System architecture For the Parameter Optimization process Using Genetic Algorithms

Table 2: The results of the proposed framework using n-gram

	Recall	Precision	F-Measure	Size of test set
Brown	97.02%	95.83%	96.42%	1003881
Inaugural	98.91%	98.75%	98.82 %	130271
ABC	97.42%	96.88%	97.15%	630004
Shakespear	93.95%	92.73%	93.34%	184194
Gutenburg	97.31%	96.50%	96.9%	1942398

were as the following; following is parameter setting of the GA. The number of chromosomes per generation is 100, the number of generations is 100, probability of crossover is 0.7, probability of mutation is 0.1 and the selection strategy is Roulette Wheel with Elitism. The best result obtained from genetic algorithm was 97.1% F-Measure when $N = 3$ and $T = 1.2$. Later, we used the optimized values of N and T to test the algorithm on the test sets. Table 3 shows the results on the tested corpora without optimization, while table 4 shows the results after applying the optimization. Similar to the results of Google n-gram, the F-measure is enhanced in all corpus.

Table 3: Expriment results on many corpora using BNC Language model of previous work [23]

	Recall	Precision	F-Measure	Size of test set
Brown	96.88%	96.72%	96.79%	1003881
Inaugural	97.8%	98.4%	98.1%	130271
ABC	95.96%	95.41%	95.68%	630004
Shakespear	94.36%	93.49%	93.92%	184194
Gutenburg	95.08%	94.37%	94.72%	1942398

There are other works in [7, 28, 19, 17] that used the BNC language model and the Brown corpus as a test set. We included there results and compared them with the result of the proposed approach. Table 4 in section 5.2 shows this comparison.

Table 4: Expriment results on many corpora using BNC Language model of the current framework

	Recall	Precision	F-Measure	Size of test set
Brown	96.96%	97.06%	97%	1003881
Inaugural	97.74%	98.84%	98.29%	130271
ABC	96.23%	96.61%	96.42%	630004
Shakespear	94.43%	94.45%	94.44%	184194
Gutenburg	95.54%	95.83%	95.68%	1942398

Table 5: Comparison between the proposed work and works in [7, 28, 19, 17]

	Kit in[19]	Peng in [28]	De Marcken in [7]	Islam in [17]	Proposed Framework
Precision	79.33%	74.6%	90.5%	89.92%	97.06%
Recall	63.01%	79.2%	17%	94.69%	96.96%
F-Measure	70.23%	75.49%	28.62%	92.24%	97%

5.3 Al-watan Language Model

The proposed method was tested on Arabic language using $10,000,000$ words of Al-watan Dataset. As usual, the dataset is split into a training set, development set and test set, with the percentage of 88%, 2% and 10% respectively. As it was mentioned previously, the training set is used to build the language model while The development is used to optimize the parameters of the proposed algorithm as depicted in figure 3. Finally, the proposed method was tested to measure the performance on the test set. The performance of the algorithm is evaluated in terms of precision, recall and f-Measure. Table 6 shows the results.

The genetic algorithm was experimented several times with different GA parameters settings. The settings that achieved the highest f-measure was selected.


These settings include the following criteria; the number of chromosomes per generation is 100, the number of generations is 100, the probability of crossover is 0.8, probability of mutation is 0.1 and selection strategy is Roulette Wheel with Elitism. The best solution in the genetic algorithm achieved 92.3% F-Measure when $N = 4$ and $T = 1.4$.

Table 6: Comparison between results before and after the optimization

	Before optimization	After optimization
Precision	86.5%	88.3%
Recall	91.5%	92.1%
F-Measure	89.1%	90.2%

6 Conclusion

In order to segment a sequence of concatenated character without spaces into meaningful words, several word segmentation techniques have been proposed. The main task of the segmentation process is to find the solution with ultimate accuracy. Usually, word segmentation techniques use heuristic methods during the segmentation process to avoid searching the unnecessary state space from one side and to choose a measure to guide the quality of the solution from the other side. This paper showed how to optimize segmentation parameters by means of genetic algorithms. In particular, the genetic algorithm has been used side by side in the segmentation process to optimize the parameter representing the length of the segmented local candidates generated during the segmentation process. Additionally, it has been used to optimize the parameter that represents the trade-off between the length of the segmented local candidates and the probability. The presented approach has been tested in two different languages, namely English and Arabic language. In the English language, the proposed approach has been experimented in both Google n-gram and BNC language model. Whereas in the Arabic language, the proposed approach was tested using El-Watan Language model. For each language model, several datasets have been taken into consideration. The proposed approach has been compared with the performance of a successful previous work without parameter optimization. The results showed that genetic algorithm has been proven to be an evolutionary approach that can be successfully applied to word segmentation.

References

[1] Mourad Abbas, Kamel Smaïli, and Daoud Berkani. Evaluation of topic identification methods on arabic corpora. *JDIM*, 9(5):185–192, 2011.

[2] Onur Boyabatli and Ihsan Sabuncuoglu. Parameter selection in genetic algorithms. *Journal of Systemics, Cybernetics and Informatics*, 4(2):78, 2004.

[3] Michael R Brent. An efficient, probabilistically sound algorithm for segmentation and word discovery. *Machine Learning*, 34(1-3):71–105, 1999.

[4] Jenna Carr. An introduction to genetic algorithms. *See: karczmarczuk. users. greyc. fr/TEACH/IAD/GenDoc/carrGenet. pdf*, 2014.

[5] Anthony Cheung, Mohammed Bennamoun, and Neil W Bergmann. An arabic optical character recognition system using recognition-based segmentation. *Pattern recognition*, 34(2):215–233, 2001.

[6] Robert Dale, Hermann Moisl, and Harold Somers. *Handbook of natural language processing*. CRC Press, 2000.

[7] Carl De Marcken. The unsupervised acquisition of a lexicon from continuous speech. *arXiv preprint cmp-lg/9512002*, 1995.

[8] Zacharias Detorakis and George Tambouratzis. Applying a sectioned genetic algorithm to word segmentation. *Pattern Analysis and Applications*, 13(1):93–104, 2010.

[9] Ágoston E Eiben, Robert Hinterding, and Zbigniew Michalewicz. Parameter control in evolutionary algorithms. *IEEE Transactions on evolutionary computation*, 3(2):124–141, 1999.

[10] Jianfeng Gao, Mu Li, Andi Wu, and Chang-Ning Huang. Chinese word segmentation and named entity recognition: A pragmatic approach. *Computational Linguistics*, 31(4):531–574, 2005.

[11] Alexander Gelbukh, Grigori Sidorov, Diego Lara-Reyes, and Liliana Chanona-Hernandez. Division of spanish words into morphemes with a genetic algorithm. In *International Conference on Application of Natural Language to Information Systems*, pages 19–26. Springer, 2008.

[12] David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

[13] David E Goldberg and H John. Holland. genetic algorithms and machine learning. *Machine learning*, 3(2-3):95–99, 1988.

[14] Julia Hockenmaier and Chris Brew. Error-driven learning of chinese word segmentation. In *12th Pacific Conference on Language and Information*, pages 218–229, 1998.

[15] John H Holland. Adaptation in natural and artificial systems. an introductory analysis with application to biology, control, and artificial intelligence. *Ann Arbor, MI: University of Michigan Press*, 1975.

[16] Nguy Thi Minh Huy, Azim Roussanaly, H Tuong Vinh, et al. A hybrid approach to word segmentation of vietnamese texts. In *Language and Automata Theory and Applications*, pages 240–249. Springer, 2008.

[17] Md Aminul Islam, Diana Inkpen, and Iluju Kiringa. A generalized approach to word segmentation using maximum length descending frequency and entropy rate. In *Computational Linguistics and Intelligent Text Processing*, pages 175–185. Springer, 2007.

[18] Dimitar Kazakov and Suresh Manandhar. A hybrid approach to word segmentation. In *International Conference on Inductive Logic Programming*, pages 125–134. Springer, 1998.

[19] Chunyu Kityz and Yorick Wilksz. Unsupervised learning of word boundary with description length gain. In *Proceedings of the CoNLL99 ACL Workshop. Bergen, Norway: Association for Computational Linguistics*, pages 1–6. Citeseer, 1999.

[20] Xiaofei Lu. Towards a hybrid model for chinese word segmentation. In *Proceedings of Fourth SIGHAN Workshop on Chinese Language Processing*, pages 189–192, 2005.

[21] Jayant Madhavan, Philip A Bernstein, AnHai Doan, and Alon Halevy. Corpus-based schema matching. In *Data Engineering, 2005. ICDE 2005. Proceedings. 21st International Conference on*, pages 57–68. IEEE, 2005.

[22] Melanie Mitchell. Genetic algorithms: An overview. *Complexity*, 1(1):31–39, 1995.

[23] Ammar Mohammed, Mohamed Karam, and Hesham Hefny. A hybrid approach for word segmentation. In *SAI Intelligent Systems Conference (IntelliSys), 2015*, pages 232–238. IEEE, 2015.

[24] Thanh V Nguyen, Hoang K Tran, Thanh TT Nguyen, and Hung Nguyen. Word segmentation for vietnamese text categorization: an online corpus approach. *RIVF06*, 2006.

[25] Peter Norvig. Natural language corpus data. *Beautiful Data*, pages 219–242, 2009.

[26] David D Palmer. Tokenisation and sentence segmentation. *Handbook of natural language processing*, pages 11–35, 2000.

[27] Eric Pellerin, Luc Pigeon, and Sylvain Delisle. Self-adaptive parameters in genetic algorithms. In *Defense and Security*, pages 53–64. International Society for Optics and Photonics, 2004.

[28] Fuchun Peng and Dale Schuurmans. A hierarchical em approach to word segmentation. In *NLPRS*, pages 475–480, 2001.

[29] Dang Duc Pham, Giang Binh Tran, and Son Bao Pham. A hybrid approach to vietnamese word segmentation using part of speech tags. In *Knowledge and Systems Engineering, 2009. KSE'09. International Conference on*, pages 154–161. IEEE, 2009.

[30] Lawrence R Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*, volume 14. PTR Prentice Hall Englewood Cliffs, 1993.

[31] Jenny R Saffran, Elissa L Newport, and Richard N Aslin. Word segmentation: The role of distributional cues. *Journal of memory and language*, 35(4):606–621, 1996.

[32] Tobias Scheidat, Andreas Engel, and Claus Vielhauer. Parameter optimization for biometric fingerprint recognition using genetic algorithms. In *Proceedings of the 8th Workshop on Multimedia and Security*, pages 130–134. ACM, 2006.

[33] Toby Segaran and Jeff Hammerbacher. *Beautiful data: the stories behind elegant data solutions.* " O'Reilly Media, Inc.", 2009.

[34] SN Sivanandam and SN Deepa. *Introduction to genetic algorithms.* Springer Science & Business Media, 2007.

[35] Richard Sproat and Thomas Emerson. The first international chinese word segmentation bakeoff. In *Proceedings of the second SIGHAN workshop on Chinese language processing-Volume 17*, pages 133–143. Association for Computational Linguistics, 2003.

[36] Bin Tan and Fuchun Peng. Unsupervised query segmentation using generative language models and wikipedia. In *Proceedings of the 17th international conference on World Wide Web*, pages 347–356. ACM, 2008.

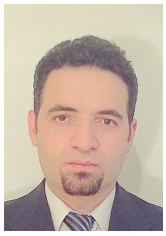
[37] William J Teahan, Yingying Wen, Rodger McNab, and Ian H Witten. A compression-based algorithm for chinese word segmentation. *Computational Linguistics*, 26(3):375–393, 2000.

[38] Pak-kwong Wong and Chorkin Chan. Chinese word segmentation based on maximum matching and word binding force. In *Proceedings of the 16th conference on Computational linguistics-Volume 1*, pages 200–203. Association for Computational Linguistics, 1996.

[39] Fong-Ching Yuan. Parameters optimization using genetic algorithms in support vector regression for sales volume forecasting. *Applied Mathematics*, 3(10):1480, 2012.

[40] Jun-Sheng Zhou, Xin-Yu Dai, Ruiyu Ni, and Jiajun Chen. A hybrid approach to chinese word segmentation around crfs. In *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*, volume 199. Jeju Island, Korea, 2005.

Biographies



Ammar Mohammed received his Bsc degree of computer science from Cairo University in 1999 and his Ms degree of computer science from the same university in 2005. He received his PhD degree of computer science from university of Koblenz-Landau, Germany in 2010; Currently he is associate professor at Department Computer Science, Institute of Statistical studies and research, Cairo University.

Mohamed Karam received his Bsc degree of commerce from Cairo University in 2010, he is currently a Computer science Master student at University of Cairo, in the ISSR institute. He is a software developer in Egypt-Soft company

Hesham Hefny received his Bsc, MSc and PhD degrees all in electronics and communication engineering from Cairo University in 1987, 1991, 1998, respectively. Currently, he is a Professor of computer science and the Head of the Department of Computer Science at the Institute of Statistical Studies and Research-Cairo University. His research of interest include fuzzy systems, artificial neural networks and granular computing